



Calhoun: The NPS Institutional Archive
DSpace Repository

Theses and Dissertations

1. Thesis and Dissertation Collection, all items

1983

An iteration algorithm for optimal network flows.

Woong, Chang Joon

Monterey, California. Naval Postgraduate School

<http://hdl.handle.net/10945/19863>

Downloaded from NPS Archive: Calhoun



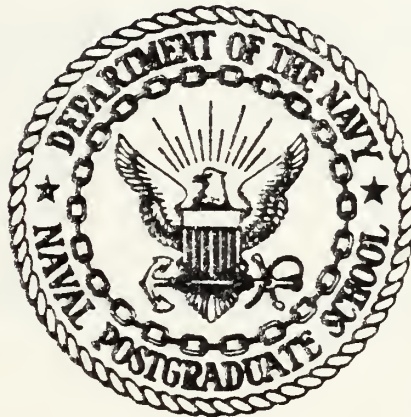
Calhoun is the Naval Postgraduate School's public access digital repository for research materials and institutional publications created by the NPS community. Calhoun is named for Professor of Mathematics Guy K. Calhoun, NPS's first appointed -- and published -- scholarly author.

Dudley Knox Library / Naval Postgraduate School
411 Dyer Road / 1 University Circle
Monterey, California USA 93943

<http://www.nps.edu/library>

NAVAL POSTGRADUATE SCHOOL

Monterey, California



THESIS

AN ITERATION ALGORITHM
FOR OPTIMAL NETWORK FLOWS

by

Chang Joon Woong

September 1983

Thesis Advisor:

J. M. Wozencraft

Approved for public release, distribution unlimited.

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) An Iteration Algorithm for Optimal Network Flows		5. TYPE OF REPORT & PERIOD COVERED Masters Thesis; September 1983
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Chang Joon Woong		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93943		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, California 93943		12. REPORT DATE September 1983
		13. NUMBER OF PAGES 84
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/ DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release, distribution unlimited.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Packet Switching, Static Routing, Optimal Flow, Conservation Constraints, Capacity Constraints, Successive Saturation, Max-slack, Delay Analysis.		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) A packet switching network has the desirable feature of rapidly handling short (bursty) messages of the type often found in computer communication systems. In evaluating packet switching networks, the average time delay per packet is one of the most important measures of performance. The problem of message routing to minimize time delay is analyzed here using two approaches, called "successive		

saturation" and "max-slack", for various traffic requirement matrices and networks with fixed topology and link capacities.

Approved for public release, distribution unlimited.

AN ITERATION ALGORITHM FOR OPTIMAL NETWORK FLOWS

by

Chang Joon Woong
Lieutenant Colonel, Korea Marine Corps
B.S., Korea Naval Academy, 1965

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL

September 1983

ABSTRACT

A packet switching network has the desirable feature of rapidly handling short (bursty) messages of the type often found in computer communication systems. In evaluating packet switching networks, the average time delay per packet is one of the most important measures of performance.

The problem of message routing to minimize time delay is analyzed here using two approaches, called "successive saturation" and "max-slack", for various traffic requirement matrices and networks with fixed topology and link capacities.

TABLE OF CONTENTS

I.	INTRODUCTION -----	10
A.	THE PACKET SWITCHED NETWORK CONCEPT -----	10
B.	THE ROUTING PROBLEM -----	13
	1. Static Routing -----	15
	a. Routing Table Representation -----	15
	b. Problem Formulation -----	16
	2. Adaptive Routing -----	17
II.	OPTIMAL FLOW AND MINIMUM CAPACITY ASSIGNMENT-----	19
A.	NETWORK MODELING -----	19
	1. Conservation Constraints -----	20
	2. Capacity Constraints -----	21
	3. Positive Constraints -----	21
B.	MODEL PROGRAMMING -----	22
	1. Node to Link Incidence Matrix -----	22
	2. Matrix Representation -----	24
	3. Programs -----	25
C.	SOLUTION TECHNIQUES -----	26
	1. Successive Saturation Approach -----	27
	2. Max-Slack Approach -----	29
D.	DELAY ANALYSIS -----	29
III.	EXPERIMENTATION RESULTS AND CONCLUSIONS -----	34
A.	EXPERIMENTS AND COMPARISON OF THE RESULTS-----	34
B.	CONCLUSIONS -----	37

APPENDIX A.	MATRIX REPRESENTATION FOR ILLUSTRATION NETWORK -----	39
APPENDIX B.	EXPERIMENTAL NETWORKS -----	40
APPENDIX C.	MODEL GENERATION PROGRAM FOR EXAMPLE PROBLEM -----	41
APPENDIX D.	OPTIMIZATION PROGRAM, INPUT DATA, MATRIX PICTURE AND OUTPUT FOR EXAMPLE PROBLEM -----	44
APPENDIX E.	OPTIMIZATION PROGRAM, INPUT DATA, MATRIX PICTURE AND OUTPUT FOR MAX-SLACK APPROACH -----	65
APPENDIX F.	LINK FLOW FOR 9/36 NETWORK -----	71
APPENDIX G.	LINK FLOW FOR 13/60 NETWORK -----	73
APPENDIX H.	LINK FLOW FOR SYMMETRIC REQ-MAT of 13/60 NETWORK -----	78
LIST OF REFERENCES	-----	83
INITIAL DISTRIBUTION LIST	-----	84

LIST OF TABLES

I.	LINK FLOW FOR THE EXAMPLE NETWORK -----	33
II.	RESULTS FOR THE 9/36 NETWORK -----	35
III.	RESULTS FOR THE 13/60 NETWORK -----	36
IV.	TIME DELAY FOR THE CAPACITIES -----	38
F.1	LINK FLOW FOR 9/36 NETWORK -----	71
G.1	LINK FLOW FOR 13/60 NETWORK -----	73
H.1	LINK FLOW FOR SYMMETRIC REQ-MAT OF 13-60 NETWORK -----	78

LIST OF FIGURES

1.1	Routing Table -----	16
2.1	The Illustration Network -----	24
3.1	The Example Network -----	28
A.1	Matrix Representation for Illustration Network--	39
B.1	The 9 Node/36 Link Network -----	40
B.2	The 13 Node/60 Link Network -----	40

ACKNOWLEDGEMENT

The author wishes to express his sincere appreciation to Professor J.W. Wozencraft for his assistance and guidance, in completing this work. Also, I want to give special thanks to my wife and family, (Chong and Hee-Won) for their combined support and understanding.

I. INTRODUCTION

A. THE PACKET SWITCHED NETWORK CONCEPT

A new technique for data communications that has evolved over 10 years is called PACKET SWITCHING. In general, communication networks may be conveniently divided into three types: CIRCUIT SWITCHING, MESSAGE SWITCHING and PACKET SWITCHING. Both message and packet switching uses a technique known as store and forward transmission.

A circuit switching network provides service by setting up a total path of connected lines from the source to the destination of the call. This complete circuit is set up by a special signaling message that threads its way through the network, seizing channels in the path as it proceeds. After the path is established, a return signal informs the source that data transmission may proceed, and all channels in the path are then used simultaneously.

The entire path remains allocated to the transmission, and only when the source release the circuit will all these channels be returned to the available pool for use in other paths. Circuit switching is the common method for telephone systems.

In message switching, only one channel is used at a time for a given transmission. The message first travels from its source node to the next node in its path, and when the

entire message is received at this node, then the next step in its journey is selected. If this selected channel (link) is busy, the message waits in a queue, and finally when the channel becomes free, transmission begins. Thus, the message "hops" from node to node through the network using only one channel at a time, possibly queueing at busy channels, as it is successively stored and forwarded through the network.

Packet switching is basically the same as message switching except that the messages are decomposed into small equal pieces called packets, each of which has a minimum length. These packets are numbered and addressed and make their way through the net independently of each other. Thus, many packets of the same message may be in transmission simultaneously, giving one of the main advantages of packet switching.

With packet switching systems, information is exchanged in the form of short packets. A packet-switched network can handle several different types of traffic concurrently. These include HIGH-THROUGHPUT traffic, for example, the transmission of large data files between computers, for which accuracy and high average data speed are the most important performance requirements; LOW-DELAY traffic, for example, interactive communication between a person at a terminal and a remote computer, for which accuracy and low average message delay are important; and REAL-TIME traffic, for example, packetized speech for which the performance of circuit-switched connection must be approached by maintaining a relatively constant

data speed, but for which extreme accuracy is not important owing to the redundancy of the information.

The packet switched network is designed primarily for computer to computer communication. It has a much more rapid response which matches the internal behavior of computers and handles information in much the same way as does a computer. At the same time it can readily match the speed of attached computers to that of the terminal users, by virtue of its internal storage.

The prime purpose of store and forward packet switching is to enable communications resources to be used effectively and in such a way that they may be shared by many users operating in an intermittent fashion, giving each user a rapid response from the communication network just at the instant when this is required.

If there is need for transmitting a long continuous stream of data, then a circuit switched connection makes good sense. On the other hand, if the data flow is bursty, then some form of resource sharing can be used to great advantage; packet switching is an effective choice here.

Since packets are stored as they pass through switching nodes, it is possible to conduct speed, format and code conversion during the switching process. Another feature of packet switching is its ability to adaptively select good paths for packet journeys as a function of the network congestion.

Besides providing small network delays, packet switching has the desirable feature of rapidly handling small messages in spite of the presence of long messages that may be in transport at the same time, this is because of the decomposition of long messages into packets. Another useful property of this decomposition is that the nodal storage requirement is reduced.

In evaluating packet switching networks, the delay, throughput, cost and reliability are important measures.

Theoretical studies have been directed to the queueing and network flow problems in general and more specifically to such problems as delay analysis, route assignment, topological design and flow control, etc.

The topic chosen in this study is centered around optimal flow and minimum capacity assignment and delay analysis for packet-switched networks. The networks under consideration have a relatively complicated structure with a large number of source-destination node pairs.

B. THE ROUTING PROBLEM

A message routing procedure is merely a decision rule that determines the node a message will next visit in its path through the network. The objective of the routing procedures is to transport packets on a minimum delay path from source to destination.

In discussing routing policies for networks, an important distinction must be made between static and adaptive policies.

This distinction depends on the environment in which a policy is designed to operate. If network topology is not subject to changes (due to failure, modifications, growth) and traffic inputs are stationary, then the optimal routing solution is a static solution consisting of a set of fixed paths between all node pairs. The traffic between each source and destination pair may be distributed on several paths simultaneously in well defined proportions, where the proportions are fixed in time.

In a real network environment, however, the topology changes with time and user traffic requirements tend to fluctuate more or less rapidly. To minimize delay it is then necessary to implement an adaptive routing policy that can react and adjust to various changes. The adaptivity of a policy can be measured in terms of its response time. A reasonable procedure is to use a periodically refreshed static routing solution. Indeed, there is a continuum of solutions between the two extremes, static and adaptive, characterized by different response times and used for different applications.

Beside their use in operational networks, static routing policies find an important application in the network design process. During this process, for a given traffic pattern a prediction of the throughput and delay performance of a given topology is needed. The routing policy clearly has a major impact on such performance. Most routing implementations are adaptive, and unfortunately the analysis of adaptive

routing policies is an extremely difficult task. To simplify the problem, the traffic pattern is usually approximated with a stationary pattern, and the routing policy with a static routing policy.

1. Static Routing

a. Routing Table Representation

A static routing policy is represented by a set of routing tables, one for each node, indicating how the traffic arriving at that node must be routed on the outgoing links, depending on its final destination. The routing table for node i is an $N \times L$ matrix $P_i(n,k)$, where N is the number of nodes in the network and $P_i(n,k)$ is the fraction of traffic directed to node k which, upon arrival at node i , is routed through neighbor node n (Fig. 1.1). By the definition,

$$P_i(j,k) = 1$$

Where n is the number of neighbors of node i .

The actual distribution of the incoming traffic to outgoing links may be done randomly using as probability weights the values $P_i(j,k)$. If for any node i there is only one permissible outgoing link from i to any destination k , then

$$P_i(j,k) = \begin{matrix} 1 & \text{for neighbor node } j. \\ 0 & \text{otherwise} \end{matrix}$$

Such a routing policy is referred to as a single path routing policy, since only one path is used from any node i to any destination node k . In general, the optimal static routing solution is a multipath solution, allowing for the

simultaneous use of several routes in order to minimize delays. The routing table representation may also be used for adaptive policies.

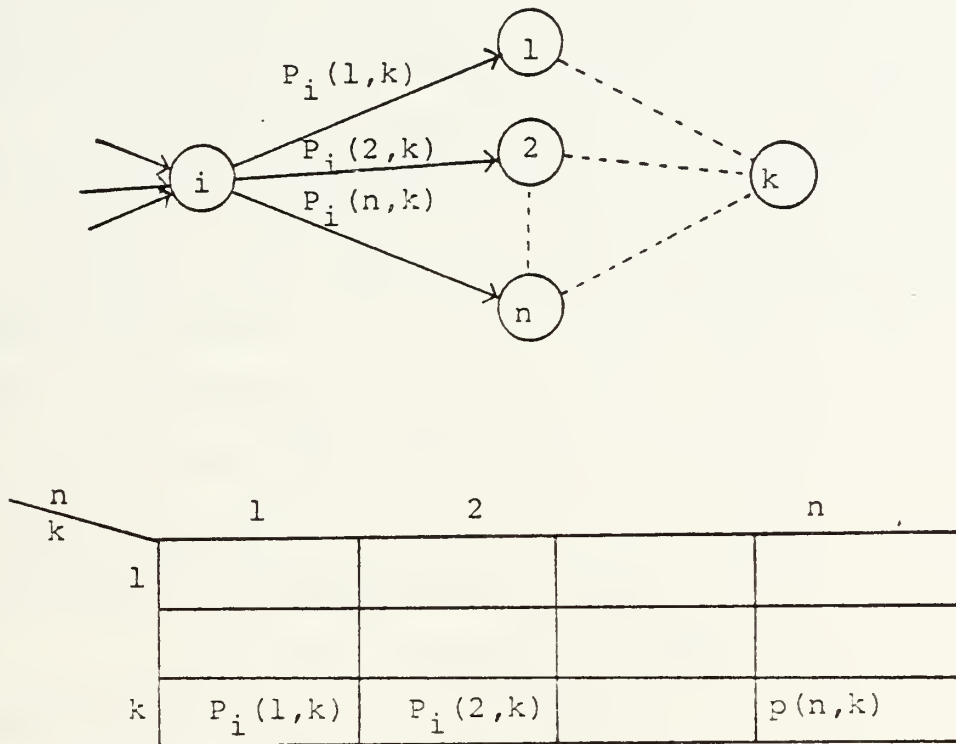


Figure 1.1. Routing Table.

b. Problem Formulation

In packet switched network, messages are segmented into small packets, and each packet travels from its source to its destination via a set of intermediate nodes. While awaiting transmission to the next node in the route, the packet must be stored until the link to that node is free. Thus, at each node there are several queues, one for each output channel.

Packet flow requirements between nodes arise at random times; therefore, link flows, queue length, and packet delay are random variables. A static routing problem can be defined as the problem of finding the static routing policy which optimizes the flow and minimizes the average time delay.

Considering the relationship between routing policy and link flows, the problem can be formulated for solution of optimal flow and minimum capacity with respect to the given requirement as follows.

given: (a) topology

(b) requirement matrix R

minimize: α

subject to: (a) conservation of flow

(b) $\text{flow} \leq \text{capacity} \times \alpha$

Where α is a nonnegative scale factor.

The mathematical model and solution techniques are discussed in detail later.

2. Adaptive Routing

The adaptive routing problem consists of defining a procedure that dynamically updates routing tables according to changes in the network. This procedure will include the acquisition of the status of all neighboring nodes, which a packet associated with the source and destination node pair could visit next after being processed at the current node. The neighboring nodes are those with direct channels linking to a given node.

The structure of the routing table and the ways the information contained therein are to be used vary according to the individual routing procedure. For example, if single path routing is used, only one path links each source and destination node pair; therefore, only a single neighboring node associated with each node pair is listed in the routing table.

The design of routing is a process of determining the structure of routing tables and specifying the procedures of using them. The choice of routing depends on many factors such as network topology, available facilities, throughput and delay requirements, and so on. Adaptive routing techniques are ruled out in this study.

II. OPTIMAL FLOW AND MINIMUM CAPACITY ASSIGNMENT

The optimal flow is the best selection of paths between source and destination, given the traffic requirements and the network configuration. The definition of best paths may vary depending on the nature of the traffic. In this study, the best paths will be regarded as the paths of minimum "distance", where distance can be interpreted in various ways, for instance as time delay along the links.

To investigate average time delay of the networks, two different approaches are discussed using classical optimization procedures.

A. NETWORK MODELING

In a distributed packet-switched network, packets are transmitted over a collection of switching nodes in tandem. In modeling a packet switched network, the objective is usually to optimize the total flow of the networks, to minimize the capacity of each link, or to maximize the utilization of the network. The problem may be formulated as follows.

given	topology
	requirement matrix R
	capacity of each link (arbitrary)
minimize:	α
subject to:	(1) conservation constraints.

$$\begin{array}{rcl}
 \sum F_{\ell}(k) & - & \sum F_{\ell}(k) = R_i(k) \\
 \text{outgoing} & & \text{incoming} \quad \text{input} \\
 \text{links} & & \text{links} \quad \text{at node } i \\
 \text{from node } i & \text{to node } i & \text{for node } k
 \end{array}$$

(2) capacity constraints

$$F_{\ell}(k) \leq \alpha \times C_{\ell}$$

(3) positive constraints

$$\sum F_{\ell}(k) \geq 0$$

Where $F_{\ell}(k)$ is the flow on link destined for node k . We next look at each constraint further in detail.

1. Conservation Constraints

At each node i , the total outgoing packet flows destined for node k equal the incoming flows plus the input to be transmitted to node k .

$$\begin{array}{rcl}
 \sum F_{\ell}(k) & - & \sum F_{\ell}(k) = R_i(k) \\
 \text{outgoing} & & \text{incoming} \quad \text{input} \\
 \text{links} & & \text{links} \quad \text{at node } i \\
 \text{from node } i & \text{to node } i & \text{for node } k
 \end{array}$$

where

k = destination node ($k = 1, 2, 3 \dots N$).

N = number of nodes.

L = number of links.

i = source node ($i = k, i = 1, 2, 3 \dots N$).

ℓ = link between two nodes ($\ell = 1, 2, 3 \dots L$).

$F_{\ell}(k)$ = flow on link destined for node k .

$R_i(k)$ = input destined for node k at node i .

2. Capacity Constraints

The total flow destined for node k on a certain link should be less than or equal to the capacity of the link ℓ . Since linear programming algorithms have difficulties dealing with inequalities, slack variables are added to the constraints to absorb unused resources and thus to force equalities to appear.

When slack variables are introduced, the problem in standard form becomes.

$$\sum_{k=1}^N F_{\ell}(k) - \alpha \times C_{\ell} + S_{\ell} = 0$$

where

C_{ℓ} = capacity of the link.

α = constant to optimally minimize the link
capacity and total flow on link.

S_{ℓ} = slack variables.

3. Positive Constraints

If we consider the network to have unidirectional links, the bidirectional flows between two nodes are divided into two one-way links in opposite direction, and the flow on each link is positive.

$$F_{\ell}(k) \geq 0$$

Using matrix notation, the model simply may be expressed as a linear programming problem.

$$\begin{array}{ll} \text{minimize:} & \alpha \\ \text{subject to:} & AX = R \\ & x \geq 0 \end{array}$$

B. MODEL PROGRAMMING

In order to use a readily available Mathematical Programming System such as MPSIII, which can solve linear programming problems with up to 4000 rows and theoretically with an unlimited number of variables, the major programming problem with large scale networks is how to generate the input data for the MPSIII program. In other words, how do we generate the matrix form of the model.

1. Node to Link Incidence Matrix

For any network processing to take place in the computer, the network structure must be represented in some machine understandable form. A wide spectrum of different representation methods are available. Most of these are based on an incidence matrix scheme. A matrix form is convenient for the analysis of networks since the resulting matrices are in a format suitable for mathematical analysis.

The network of Fig. 2.1 will be used to illustrate the method. The node to link incidence matrix $E(i, \ell)$ describes the links connected to the node such that

$$E(i, \ell) = \begin{cases} 1 & \text{if link } \ell \text{ is outgoing from node } i. \\ -1 & \text{if link } \ell \text{ is incoming to node } i. \\ 0 & \text{if link } \ell \text{ is not connected to node } i. \end{cases}$$

The incidence matrix for input at node i destined for node k is a modification of the node to link incidence matrix $E(i, \ell)$ according to the following rules:

$$E_k(i, \ell) = \begin{cases} 0 & \text{for columns corresponding to links} \\ & \text{outgoing from node } k. \\ \text{delete the row if } i = k & \\ E(i, \ell) & \text{otherwise} \end{cases}$$

For the illustration network (Fig. 2.1).

		ℓ		1	2	3	4	5
		i						
$E(i, 1) = 1$		1		1	-1	1	-1	0
		2		-1	1	0	0	-1
		3		0	0	-1	1	1

for $k = 1$

		ℓ		1	2	3	4	5
		i						
$E_1(i, 1) = 2$		2		0	1	0	0	-1
		3		0	0	0	1	1

$E_2(i, \ell)$ and $E_3(i, \ell)$ may be represented in the same manner.

for $k = 2$

		ℓ		1	2	3	4	5
		i						
$E_2(i, 1) = 1$		1		1	-1	1	-1	0
		3		0	0	-1	1	1

for $k = 3$

		ℓ		1	2	3	4	5
		i						
$E_3(i, \ell) = 1$		1		1	-1	1	-1	0
		2		-1	1	0	0	-1

$E(i,\ell)$ is a 3×5 matrix and $E_k(i,\ell)$ is a 2×5 matrix in size.

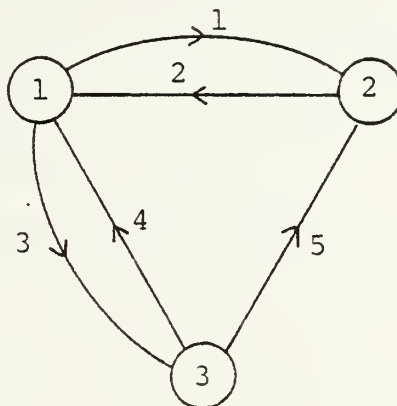


Figure 2.1. Illustration Network.

2. Matrix Representation

Matrix representation is a convenient tool for programming. The complete matrix representation for the illustration network is shown in Appendix A.

The general model matrix $AX = R$ may be represented as follows:

$$\begin{array}{c}
 \begin{array}{c} A \\
 \left| \begin{array}{cccccc}
 E1 & & & & & \\
 & E2 & & & & \\
 & & E3 & & & \\
 & & & \cdot & & \\
 & & & & \cdot & \\
 & & & & & \cdot \\
 & & & & & E_k \\
 I1 & I2 & I3 & \cdot & \cdot & \cdot & I_k
 \end{array} \right| - C^l
 \end{array}
 \end{array}
 =
 \begin{array}{c}
 \begin{array}{c} X \\
 \left| \begin{array}{c}
 F^l(1) \\
 F^l(2) \\
 F^l(3) \\
 \cdot \\
 \cdot \\
 \cdot \\
 F^l(k) \\
 F^l(k) \\
 S^l \\
 \text{Alpha}
 \end{array} \right|
 \end{array}
 \end{array}
 =
 \begin{array}{c}
 \begin{array}{c} R \\
 \left| \begin{array}{c}
 R_i(1) \\
 R_i(2) \\
 R_i(3) \\
 \cdot \\
 \cdot \\
 \cdot \\
 R_i(k) \\
 0 \rangle \quad L \\
 \cdot \\
 0 \rangle
 \end{array} \right|
 \end{array}
 \end{array}$$

where

E_k = $(N-1) \times L$ incident matrix.

I_k = $L \times L$ identity matrix with zero elements
corresponding to zero columns of the E_k matrix.

I = $L \times L$ identity matrix.

$R_i(k)$ = requirement matrix which is the inputs to be
sent from source node i to destination node k .

The size of the model in matrix representation is;

$A = ((N-1) \times N + L) \times ((N+1) \times L + 1)$

$X = ((N+1) \times L + 1) \times 1$

$R = ((N-1) \times N + L) \times 1$

For example, the network consisting of 20 nodes and
40 links has the matrix which is $A = 420 \times 841$, $X = 841 \times 1$
and $R = 420 \times 1$ in size.

3. Programs

To generate the MPSIII input data and to solve the
model, requires two programs which are listed at the end of
this study (Appendix C and D).

The first one is the data (model) generation program
(Appendix C) written in the Fortran language. The output of
this program uses special notation to designate the flow of
the link connecting node pairs, destination node of the flow,
flow variables, slack variables and input at the node. These
notations are composed of a letter which is one of C, L, X or
S and 7 numerical digits.

The notation C_l , L_l , X_l , and S_l imply conservation constraints, capacity constraints, flow variables, and slack variables respectively. The rest of the six numbers following C_l , L_l , X_l and S_l indicates link connecting node pair with first 4 digits and destination node or slack variable number with last 2 digits. For example, the notation $C1000201$ indicates the conservation constraint row which is the input at node 2 destined for node 1, $L1020100$ the link capacity row for the link connecting between node 2 and node 1, $X1020101$ the flow variable on the link connecting between node 2 and node 1 destined for node 1 and $S1000001$ the first slack variable.

The second program is the MPSIII program developed by Management Science Systems, Inc. It provides elaborate control language for the formulation of solution strategies for mathematical programming problems.

Individual instructions in this language bring in quite elaborate sections of code designed to execute the step as efficiently as possible.

Using the special notation discussed above, there is a limitation on the size of the problem which can be solved, because data names must consist of 8 characters maximum. This means that the maximum network size must be less than 100 nodes, if the computer capacity allows.

C. SOLUTION TECHNIQUES

As mentioned before, the classical optimization procedures to solve the investigation model provide solutions

which are the optimal paths and minimum link capacities for the network. Different optimal solutions may be expected depending on different methods of optimization. The two different approaches, each involving iteration of the linear programming solution procedure are discussed below, and illustrated in terms of one example network problem. These approaches are based on the characteristics of linear programming problems.

1. Successive Saturation Approach

The standard procedure of the MPSIII program listed at the end of this study (Appendix D and E) produces an "optimal" solution for alpha from the output data of the data generation program in the first run. But the corresponding flow solution is not a good solution in terms of the average time delay in the packet-switched network, when it is analyzed by the Kleinrock's delay analysis model (Ref. 4). The problem is that only the flow thru the saturated links is optimum.

In order to reduce the time delay, Iteration runs are attempted until all links are saturated. The detailed iteration method for the example network (Fig. 3.1) which has 5 nodes and 6 links, each with original link capacities 10 (to simplify the problem) is discussed below.

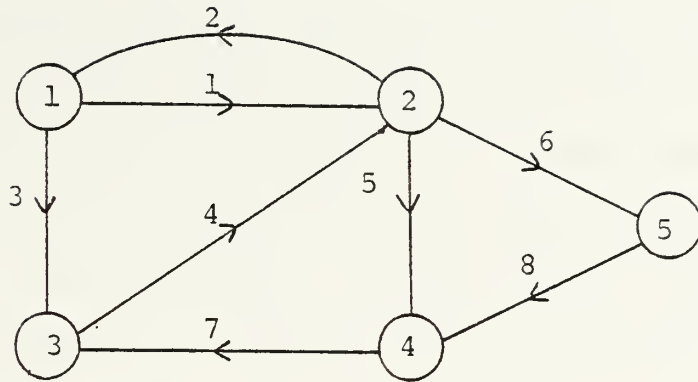


Figure 3.1. Example Network.

The output of MPSIII program is divided into row section and column section (Ref. 9 and Appendix D and E). The objective function value (which is the same as the alpha value) and the links saturated with the alpha value in the capacity constraint rows indicated with the first letter notation L can be read, and the flows on each link and the alpha value in column section. In the example network, for the first iteration the saturation level (alpha value) is 0.25, and the saturated links are L1010300 (link 3) and L1040300 (link 7). Their activity and slack activity levels are all zero, but their dual activity level is not (pp 48).

For the next iteration, the changed input datas are that the original capacity multiplied by the saturation level of first run are moved from the alpha column to the right hand side (RHS) column. In the example, the new link capacities 2.5 (0.25×10) are moved to RHS (pp. 50), and the problem is solved again. Repeating this iteration procedures until all links are saturated and moved to RHS, the objective

function value becomes zero. When the objective function value is zero, the final flow solution is the desired solution of this successive saturation approach. In the example case, the six iterations are required to solve the problem. These whole procedures except programs and input datas from 2nd run for the example network are listed at the end of this study (Appendix D).

2. Max-Slack Approach

To simplify the solution of the above successive saturation approach, another approach called the max-slack approach is considered. It requires only two iteration of the program, but is otherwise quite similar. Like in the successive saturation approach, after the first run, all saturated link capacities are multiplied by the alpha value of the first run are moved to the RHS, and in addition, all slack variable columns are summed as the objective function in the data card for the second run.

In the example network, eight slack column variables are added on the objective rows, and all new capacities 2.5 (0.25×10) are moved to RHS. This is clearly shown on the output of control language "picture", and the program, input data and the output are also listed at the end (Appendix E).

D. DELAY ANALYSIS

The average time delay of a network is the average time a packet spends in the network traveling from its source to its destination nodes. To obtain a tractable expression for

average time delay T in terms of the capacity of link ℓ , it is necessary to make simplifying assumptions. Packet lengths are assumed to be exponentially distributed, and are re-chosen with statistical independence at each node to form a Poisson process.

These assumptions known as the 'Independence assumption', introduced by D. Kleinrock (Ref. 4), make each queuing problem independent. In the analysis, each queue in a packet network is assumed to be an M/M/1 system with arrival rate of G_ℓ and to be independent of each other.

The average delay for link ℓ is given by the waiting time in the queue as

$$T_\ell = \frac{1}{uC_\ell - G_\ell}$$

Where G_ℓ is the packet traffic in the link and $1/u$ the average packet length in bits. To form a suitable average over all the queuing processes, the T are weighed by G_ℓ/r where r is the total input packet rate to the network. In this way, the total delay suffered by all packets per second of network operation $\sum_\ell G_\ell \cdot T$ is divided by the total number of packets carried by the network per second.

The total average time delay per packet through the network is then given by

$$T = \frac{1}{r} \sum_{\ell=1}^L \frac{G_\ell}{uC_\ell - G_\ell} = \frac{1}{r} \sum_{\ell=1}^L \frac{F_\ell}{C_\ell - F_\ell}$$

Where $G\ell/u = F\ell$, and L is the total number of links in the system and l/u is equal to the average flow on link ℓ . It should be noted that the value of $C\ell$ (the minimum required capacity) must be greater than the total of all the flows on link ℓ . In other words, the value of $C\ell$ must be greater than the value which is the first run saturation level multiplied by the arbitrary capacity C for the first run.

The value of T includes both of the waiting time and the service time. The waiting time is subject to the interference of all other traffic within the network that consists of the data traffic as well as the control traffic. This average time delay can be used to analyze the relative goodness of the routing strategies considered in this study.

In deriving this basic formula, a number of factors (such as processing time and propagation delay) are neglected. In any realistic network, these variables as well as others must be considered in the analysis of the networks. According to the above formula, the average time delay depends only on the aggregate flow of the links. The reduction of the flow $F\ell$ of each link results in the reduction of the average time delay for the network which has constant input packet rate and link capacities.

For the example network (Fig. 3.1), the arbitrary capacity is 10 and the value of $C\ell$ must be greater than 2.5 (0.25×10). The average time delay for the capacity $F\ell=10$, and the input (requirement matrix) of 2 units from node 1 to

node 4 and 5 units from node 2 to node 3 are $T_{ss} = 2.12/r$ for the successive saturation case and $T_{ms} = 2.083/r$ for the max-slack case. To make these calculations easy, the data summarizing the final output run are listed in Table I. We see that there is not much advantage of one procedure over the other in terms of performance, whereas the max-slack solution requires only two iterations, rather than 6 for the successive saturation algorithm.

TABLE I
LINK FLOW FOR THE EXAMPLE NETWORK

(); Max-Slack

LINK (Nodes)	DESTINATION NODE					TOTAL FLOW
	1	2	3	4	5	
1-2				(2) 2		(2) 2
2-1		(2.5) 2.5				(2.5) 2.5
1-3		(2.5) 2.5				(2.5) 2.5
3-4						
2-4		(2.5) 2.25				(2.5) 2.25
2-5		0.25		(2) 2		(2) 2.25
4-3		(2.5) 2.5				(2.5) 2.5
5-4		0.25		(2) 2		(2) 2.25
Req Mat		$C=10, R_1(4)=2, R_2(3)=5$				

III. EXPERIMENTAL RESULTS AND CONCLUSIONS

A. EXPERIMENTS AND COMPARISON OF THE RESULTS

The experiment was conducted for two experimental networks to analyze both approaches, in addition to the example network. These networks are specified in Appendix B. The networks were used in the experiment with requirement matrices of 15 units from node 1 destined for node 5, 9 units from node 7 destined for node 2 for the 9 nodes/36 links network (Fig. B.1): and 15 units from node 2 destined for node 12, 4 units from node 5 destined for node 6, and 10 units from node 10 destined for node 1 for the 13 nodes/60 links network (Fig. B.2). Both the successive saturation and max-slack approaches were used for the successive saturation approach, the 21 iterations are required in this experiment.

The results and their comparison is listed in Tables II, III, and IV. These are obtained by writing each flow variable value from the final output run on each link of the networks, and summing these flow variable values to obtain the total flow on the link (Appendix F and G).

As shown in Tables II and III, the number of iterations required and the total link usage (utilization) are much greater for the successive saturation approach, but fewer large link capacities are required than in the max-slack approach. The average time delay depends on the capacity

TABLE II
RESULTS FOR THE 9/36 NETWORK

Fℓ		4	3.75	3	2.25	1.88	1.75	1.63	1	0.5	TOTAL LINK USED	ITER REQ
MIN-CAP		4	3.75	3	2.25	1.88	1.75	1.63	1	0.5		
SUCC												
NUM	-	8	4		4	4	2	2		2	26	10
	SAT											
OF												
LINKS	MAX											
	-											
	SLACK	14		2					2		18	2

TABLE III
RESULTS FOR THE 13/60 NETWORK

$F\ell$ MIN-CAP	3.75	3.33	2.92	2.75	2.6	2.5	2.36	2.08	2	1.97	1.88
NUM	8	3	6		4		3	1	2	5	3
OF	SUCC - SAT										
LINKS	MAX - SLACK	23		1		3					
$F\ell$ MIN-CAP	1.56	1.39	1.3	1.17	0.63	0.47	0.7	0.25	TOT LINK USED	ITER REQ	
NUM	2	1	2	3	2	1	2	48	21		
OF	SUCC - SAT										
LINKS	MAX - SLACK							5	32	2	

value C_l (Table IV). When the capacity value C_l is close to the minimum required capacity, the time delay is less for the successive saturation approach than for the max-slack approach in both networks, and vice versa. It is significant for the large scale network, and for large enough link capacities C_l , that the time delay difference becomes negligible.

Also, in the process of this study, it was noted that both approaches yield the same aggregate flows in the case of very simple networks (nodes less than or equal to 4), and for completely symmetric networks and requirement matrices (input and capacity matrices). For example, the time delays are the same for both approaches with 2 units input from the outmost nodes to the directly opposite nodes respectively (1 to 13, 2 to 12, 5 to 9, 12 to 2 and 13 to 1) for the 13 node network, because all links total flow are $3/4$. The detail experimental results are listed in Appendix H.

B. CONCLUSIONS

This is an elementary study of the routing design problem for a packet network: given a traffic requirement matrix, minimize the average time delay per packet, subject to finding a feasible flow for a network with fixed topology and link capacities.

The general performance characteristics and advantages of the two approaches are investigated for small and simple networks. This illustrates one way to solve the design problem under limited conditions.

TABLE IV
TIME DELAY FOR THE CAPACITIES

CAPACITY	4	4.2	4.5	5	10	15
9/36	SUCC	244.1/r	109.5/r	59.9/r	10.74/r	5.82/r
	-					
	SAT					
Net	MAX					
	-	28.56/r	116.6/r	59.5/r	10.41/r	5.73/r
	SLACK					
13/60	SUCC	169.9/r	115.6/r	75.7/r	60.1/r	15.84/r
	-					
	SAT					
Net	MAX					
	-	335.7/r	188.3/r	114.2/r	69.3/r	14.29/r
	SLACK					

APPENDIX A
MATRIX REPRESENTATION FOR ILLUSTRATION NETWORK

A	X	=	R
0 1 0 0 1	F ₁ (1)		R ₂ (1)
0 0 0 1 1	F ₂ (1)	=	R ₃ (1)
1 0 1 -1 0	F ₃ (1)		R ₁ (2)
0 0 -1 1 1	F ₄ (1)		R ₃ (2)
1 -1 1 0 0	F ₅ (1)		R ₁ (3)
-1 1 0 0 0	F ₁ (2)		R ₂ (3)
0 1 1 1 -C1	F ₂ (2)		0
1 0 1 1 -C2	F ₃ (2)		0
0 1 1 1 -C3	F ₄ (2)		0
1 1 0 1 -C4	F ₅ (2)		0
1 1 0 1 -C5	F ₁ (3)		0
	F ₂ (3)		
	F ₃ (3)		
	F ₄ (3)		
	F ₅ (3)		
	S ₁		
	S ₂		
	S ₃		
	S ₄		
	S ₅		
	ALPHA		

Figure A.1. Matrix Representation for Illustration Network.

APPENDIX B

EXPERIMENTAL NETWORKS

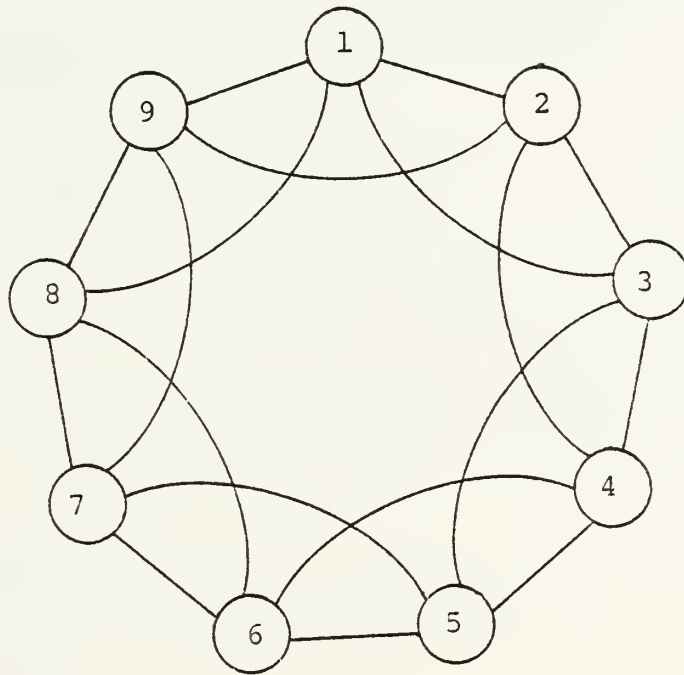


Figure B.1. The 9 Node/36 Link Network.

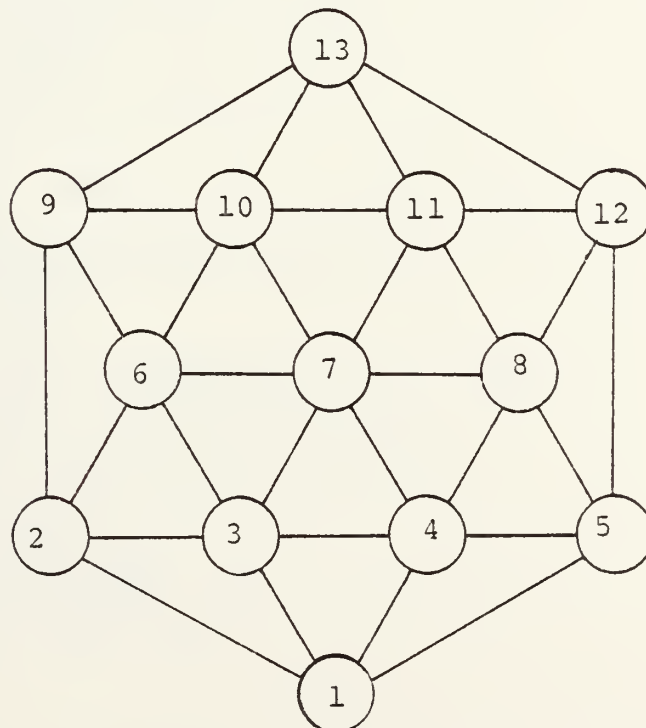


Figure B.2. The 13 Node/60 Link Network.

APPENDIX C MODEL GENERATION PROGRAM FOR EXAMPLE PROBLEM

```

$JOB          1538P
C$OPTIONS FREE
C THIS IS THE DATA GENERATION PROGRAM FOR MPSIII.
C THE FORMAT SHOULD BE CHANGED ACCORDINGLY.
C N IS NUMBER OF NODES, L IS NUMBER OF LINKS.
C IE(I,L) IS DEFINED AS FOLLOWS:
C   1. LINK INCOMING TO NODE I IS -1.
C   2. LINK OUTGOING FROM NODE I IS 1.
C   3. LINK UNCONNECTED WITH NODE I IS 0.
C IZCOL(K,L) IS DEFINED AS FOLLOWS:
C   1. LINK OUTGOING FROM NODE K IS 0.
C   2. OTHERWISE, 1.
C IR(I,K) IS THE INPUT MATRIX FROM NODE I TO NODE K.
C CAP(L,1) IS THE LINK CAPACITY MATRIX.
C COLN1(1,J1) IS THE LINK MATRIX CONNECTING NODES.
C
C   DIMENSION INC(13,60),IE(13,60),IZCOL(13,60),
C   *   IRN(13,60),IR(13,13),CAP(60,1),ICOLN1(1,60),
C   *   ICOLN(13,60),ICOLN2(13,60)
C   *   INTEGER I,J,K,L,N,NULL,IDEN,KA,J1,INC2,IA,
C   *   ID,IQ,IROWNG,IL,KC,KD,IRN1
C
C *** DATA INITIALIZATION *****
C   READ(5,200) N,L
C   READ(5,225)((ICOLN1(1,J1),J1=1,L)
C   READ(5,220)((IE(I,J),I=1,N),J=1,L)
C   READ(5,230)((IR(I,K),I=1,N),K=1,N)
C   READ(5,240)(CAP(IQ,1),IQ=1,L)
C   200 FORMAT(2I5)
C   220 FORMAT(5I3)
C   225 FORMAT(4I10)
C   230 FORMAT(5I3)
C   240 FORMAT(4E12.5)
C
C *** ZERO COLUMN GENERATION ****
C   DO 19 J = 1,L
C   DO 20 I = 1,N
C   INCE = IE(I,J)
C   IF(INCE .EQ. 1) GO TO 21
C   INCE = 1
C   GO TO 23
C 21 INCE = 0
C 23 IZCOL(I,J)=INCE
C 20 CONTINUE
C 19 CONTINUE
C *** COLUMN NUMBER GENERATION ****
C   IAD = 1000000
C   DO 1 K = 1,N
C   IROWNG = 0
C   DO 12 J1 = 1,L
C   ICOLN(K,J1) = ICOLN1(1,J1) + K + IAD
C   ICOLN2(K,J1) = ICOLN(K,J1) - K
C 12 CONTINUE
C *** INCIDENT MATRIX GENERATION ***
C   DO 2 I = 1,N
C   IF(I .EQ. K) GO TO 2
C   DO 3 J = 1,L
C   NULL = IZCOL(K,J)
C   IF(I .EQ. K) NULL = 0
C   IDEN = IE(I,J)
C   INC(I,J) = NULL*IDEN
C 3 CONTINUE

```



```

      IRDWN0 = I*100
      IRN(I,K) = IRCWNO + K + IAD
2    CONTINUE
C *** ROW SECTION GENERATION ***
      ID = 1
      DO 61 I = 1,N
      IF(I.EQ.K) GO TO 61
      WRITE(6,140) IRN(I,K)
61   CONTINUE
1    CONTINUE
C *** CAPACITY ROW GENERATION ***
      DO 7 J = 1,L
      WRITE(6,145) ICOLN2(1,J)
7    CONTINUE
C *** COLUMN SECTION GENERATION ***
      WRITE(6,500)
      DO 65 K = 1,N
      DO 62 J = 1,L
      NA = 0
      DO 63 I = 1,N
      IF(I.EQ.K) GO TO 63
      NULL = IZCJL(K,J)
      IDEN = IE(I,J)
      INC(I,J) = NULL*IDEN
      INC2 = INC(I,J)
      IF(INC2.EQ.0) GO TO 63
      NA = NA + 1
      WRITE(6,510) ICOLN(K,J), IRN(I,K), INC2
63   CONTINUE
      IF(NA.EQ.0) GO TO 62
      WRITE(6,520) ICOLN(K,J), ICOLN2(K,J), ID
62   CONTINUE
65   CONTINUE
C *** CAPACITY GENERATION ***
      DO 66 J = 1,L
      IA = J + 1000000
      WRITE(6,540) IA, ICOLN2(1,J), ID
66   CONTINUE
C *** ALPHA COLUMN GENERATION ***
      WRITE(6,565) ID
      DO 67 J = 1,L
      WRITE(6,570) ICOLN2(1,J), CAP(J,1)
67   CONTINUE
C *** RHS GENERATION ***
      WRITE(6,550)
      DO 69 K = 1,N
      DO 68 I = 1,N
      IF(I.EQ.K) GO TO 68
      WRITE(6,560) IRN(I,K), IR(I,K)
68   CONTINUE
69   CONTINUE
      DO 5 IQ=1,L
5    CONTINUE
      WRITE(6,580)
140  FORMAT(1X,'E C',I7)
145  FORMAT(1X,'E L',I7)
500  FORMAT(' COLUMNS')
510  FORMAT(4X,'X',I7,2X,'C',I7,2X,I5)
520  FORMAT(4X,'X',I7,2X,'L',I7,2X,I5)
540  FORMAT(4X,'S',I7,2X,'L',I7,2X,I5)
550  FORMAT(' RHS')
560  FORMAT(4X,' INPUT ',5X,'C',I7,2X,I5)
565  FORMAT(4X,' ALPHA ',5X,'OBJ',7X,I5)
570  FORMAT(4X,' ALPHA ',5X,'L',I7,2X,E12.5)
580  FORMAT(' ENDATA')
      STOP
      END

```


SENTRY

APPENDIX D

OPTIMIZATION PROGRAM, INPUT DATA MATRIX PICTURE AND OUTPUT FOR EXAMPLE PROBLEM

```
//CHANG JOB (1538,1808),'CHANG JOON WONG',CLASS=A
/**      MPSIII EXAMPLE PROBLEM
/**
//      EXEC      MSSMPS
//CPC.SYSIN DD *
          PROGRAM ('ND')
          INITIALZ
          TITLE ('EXAMPLE PROBLEM, 1ST RUN FOR SUCC-SAT')
          MOVE     (XOBJ,'OBJ')
          MOVE     (XRHS,'INPUT')
          MOVE     (XDATA,'LINEQS')
          MOVE     (XPBNAME,'PROJECT')
          CONVERT  ('SUMMARY')
          SETUP   ('MIN')
          BCDOUT
          PICTURE
          WHIZARD
          PRIMAL
          SOLUTICN
          EXIT
          PEND
```

```
/*
//EXEC.SYSIN      DD *
NAME              LINEQS
ROWS
```

```

N      OBJ
M      C1000201
M      C1000301
M      C1000401
M      C1000501
M      C1000102
M      C1000302
M      C1000402
M      C1000502
M      C1000103
M      C1000203
M      C1000403
M      C1000503
M      C1000104
M      C1000204
M      C1000304
M      C1000504
M      C1000105
M      C1000205
M      C1000305
M      C1000405
M      L1010200
M      L1020100
M      L1010300
M      L1030200
M      L1020400
M      L1020500
M      L1040300
M      L1050400
```

```
COLUMNS
X1020101      C1000201      1
X1020101      L1020100      1
X1030201      C1000201      -1
X1030201      C1000301      1
X1030201      L1030200      1
X1020401      C1000201      1
X1020401      C1000401      -1
X1020401      L1020400      1
```


X1020501	C1000201	1
X1020501	C1000501	-1
X1020501	L1020500	1
X1040301	C1000301	-1
X1040301	C1000401	1
X1040301	L1040300	1
X1050401	C1000401	-1
X1050401	C1000501	1
X1050401	L1050400	1
X1010202	C1000102	1
X1010202	L1010200	1
X1010302	C1000102	1
X1010302	C1000302	-1
X1010302	L1010300	1
X1030202	C1000302	1
X1030202	L1030200	1
X1040302	C1000302	-1
X1040302	C1000402	1
X1040302	L1040300	1
X1050402	C1000402	-1
X1050402	C1000502	1
X1050402	L1050400	1
X1010203	C1000103	1
X1010203	C1000203	-1
X1010203	L1010200	1
X1020103	C1000103	-1
X1020103	C1000203	1
X1020103	L1020100	1
X1010303	C1000103	1
X1010303	L1010300	1
X1020403	C1000203	1
X1020403	C1000403	-1
X1020403	L1020400	1
X1020503	C1000203	1
X1020503	C1000503	-1
X1020503	L1020500	1
X1040303	C1000403	1
X1040303	L1040300	1
X1050403	C1000403	-1
X1050403	C1000503	1
X1050403	L1050400	1
X1010204	C1000104	1
X1010204	C1000204	-1
X1010204	L1010200	1
X1020104	C1000104	-1
X1020104	C1000204	1
X1020104	L1020100	1
X1010304	C1000104	1
X1010304	C1000304	-1
X1010304	L1010300	1
X1030204	C1000204	-1
X1030204	C1000304	1
X1030204	L1030200	1
X1020404	C1000204	1
X1020404	L1020400	1
X1020504	C1000204	1
X1020504	C1000504	-1
X1020504	L1020500	1
X1050404	C1000504	1
X1050404	L1050400	1
X1010205	C1000105	1
X1010205	C1000205	-1
X1010205	L1010200	1
X1020105	C1000105	-1
X1020105	C1000205	1
X1020105	L1020100	1
X1010305	C1000105	1

X1010305	C1000305	-1	
X1010305	L1010200	-1	
X1030205	C1000205	-1	
X1030205	C1000305	-1	
X1030205	L1030200	-1	
X1020405	C1000205	-1	
X1020405	C1000405	-1	
X1020405	L1020400	-1	
X1020505	C1000205	-1	
X1020505	L1020500	-1	
X1040305	C1000305	-1	
X1040305	C1000405	-1	
X1040305	L1040300	-1	
S1000001	L1010200	-1	
S1000002	L1020100	-1	
S1000003	L1010300	-1	
S1000004	L1030200	-1	
S1000005	L1020400	-1	
S1000006	L1020500	-1	
S1000007	L1040300	-1	
S1000008	L1050400	-1	
ALPHA	OBJ	-1	
ALPHA	L1010200	-0.100000E	02
ALPHA	L1020100	-0.100000E	02
ALPHA	L1010300	-0.100000E	02
ALPHA	L1030200	-0.100000E	02
ALPHA	L1020400	-0.100000E	02
ALPHA	L1020500	-0.100000E	02
ALPHA	L1040300	-0.100000E	02
ALPHA	L1050400	-0.100000E	02
RHS			
INPUT	C1000201	0	
INPUT	C1000301	0	
INPUT	C1000401	0	
INPUT	C1000501	0	
INPUT	C1000102	0	
INPUT	C1000302	0	
INPUT	C1000402	0	
INPUT	C1000502	0	
INPUT	C1000103	0	
INPUT	C1000203	5	
INPUT	C1000403	0	
INPUT	C1000503	0	
INPUT	C1000104	2	
INPUT	C1000204	0	
INPUT	C1000304	0	
INPUT	C1000504	0	
INPUT	C1000105	0	
INPUT	C1000205	0	
INPUT	C1000305	0	
INPUT	C1000405	0	
ENDATA			
/*			
//			

EXAMPLE PROBLEM, 1ST RUN FOR SUCCESSIVE SAT

SECTION 1 - FCWS

NUMBER	...	FCW...	AT	...ACTIVITY...	SLACK ACTIVITY	...LOWER LIMIT...	...UPPER LIMIT...	...EUAL ACTIVITY
A	1	DB	BS	.25000	.25000-	NONE	NONE	1.00000
A	2	CC	DE
A	3	CC	EE
	4	CC	EE
	5	CC	EE
	6	CC	FS
	7	CC	FS
	8	CC	BS
	9	CC	BS
	10	CC	BS
	11	CC	DE	5.00000	.	5.00000	5.00000	.00000-
	12	CC	DE00000-
	13	CC	DE00000-
	14	CC	DE	2.00000	.	2.00000	2.00000	.00000-
	15	CC	DE
	16	CC	DE
	17	CC	DE
	18	CC	DE
	19	CC	DE
	20	CC	DE
	21	CC	DE
	22	CC	DE
	23	CC	DE
	24	CC	DE
	25	CC	DE
	26	CC	DE
	27	CC	DE
	28	CC	DE
	29	CC	DE

EXAMPLE PROBLEM, 1ST RUN FOR SUCCESSIVE SAT

SECTION 2 - COLUMNS

NUMBER	C COLUMN	AT	...ACTIVITY%..	..INPUT CCST..	..LOWER LIMIT.	..UPPER LIMIT.	..REDUCED COST.
30	X1C01C1	BS	.	.	.	NONE	.
31	X1C02C1	BS	.	.	.	NONE	.C5000
32	X1C03C1	LL	.	.	.	NONE	.C5000
33	X1C04C1	LL	.	.	.	NONE	.
34	X1C05C1	LL	.	.	.	NONE	.
35	X1C06C1	BS	.	.	.	NONE	.
36	X1C07C2	LL	.	.	.	NONE	.
37	X1C08C2	LL	.	.	.	NONE	.C5000
38	X1C09C2	LL	.	.	.	NONE	.C5000
39	X1C10C2	LL	.	.	.	NONE	.
40	X1C11C3	LL	.	.	.	NONE	.
41	X1C12C3	BS	.	.	.	NONE	.
42	X1C13C3	BS	2.C0000	.	.	NONE	.
43	X1C14C3	BS	2.C0000	.	.	NONE	.
44	X1C15C3	BS	2.C0000	.	.	NONE	.
45	X1C16C3	BS	2.C0000	.	.	NONE	.
46	X1C17C3	BS	2.C0000	.	.	NONE	.
47	X1C18C3	BS	2.C0000	.	.	NONE	.
48	X1C19C4	BS	2.C0000	.	.	NONE	.
49	X1C20C4	LL	.	.	.	NONE	.C5000
50	X1C21C4	LL	.	.	.	NONE	.
51	X1C22C4	BS	.	.	.	NONE	.
52	X1C23C4	BS	2.C0000	.	.	NONE	.
53	X1C24C4	BS	2.C0000	.	.	NONE	.
54	X1C25C5	BS	.	.	.	NONE	.
55	X1C26C5	BS	.	.	.	NONE	.C5000
56	X1C27C5	BS	.	.	.	NONE	.C5000
57	X1C28C5	LL	.	.	.	NONE	.
58	X1C29C5	LL	.	.	.	NONE	.
59	X1C30C5	BS	5.C000	.	.	NONE	.
60	X1C31C5	BS	.	.	.	NONE	.
61	X1C32C5	LL	.	.	.	NONE	.C5000
62	X1C33C5	LL	.	.	.	NONE	.
63	X1C34C5	LL	2.C0000	.	.	NONE	.
64	X1C35C5	BS	5.C000	.	.	NONE	.
65	X1C36C5	BS	.	.	.	NONE	.
66	X1C37C5	LL	.	.	.	NONE	.
67	X1C38C5	LL	.	.	.	NONE	.C5000
68	X1C39C5	LL	.	.	.	NONE	.
69	X1C40C5	BS	.	.	.	NONE	.
70	ALL	BS	25.C00	1.C0000	.	NONE	.

EXAMPLE PROBLEM, 2ND RUN FOR SUCCESSIVE SAT

SECTION 1 - FCWS

NUMBER	FCW	AT	ACTIVITY...	SLACK ACTIVITY	LOWER LIMIT	UPPER LIMIT	DUAL ACTIVITY
1	0E	BS	.25000	.25000-	NONE	NONE	1.00000
2	01	EQ10000-
3	02	EQ10000-
4	03	FS10000-
5	04	FS10000-
6	05	FS
7	06	FS
8	07	FS
9	08	FS
10	09	FS
11	10	FS	5.00000	.	5.00000	5.00000	.10000-
12	11	FS10000-
13	12	FS10000-
14	13	FS	2.00000	.	2.00000	2.00000	.10000-
15	14	FS
16	15	FS10000
17	16	FS
18	17	FS
19	18	FS10000-
20	19	FS10000
21	20	FS10000
22	21	FS
23	22	FS	2.50000	.	2.50000	2.50000	.10000
24	23	FS
25	24	FS
26	25	FS
27	26	FS
28	27	FS	2.50000	.	2.50000	2.50000	.10000
29	28	FS

EXAMPLE PROBLEM, 2ND RUN FOR SUCCESSIVE SAT

SECTION 2 - COLUMNS

NUMBER	COLUMN.	AT	...ACTIVITY...	..INPUT CCST..	..LOWER LIMIT.	..UPPER LIMIT.	..REDUCED COST.
30	X1C2C1C1	BS	.	.	NONE	NONE	.
31	X1C2C2C1	BS	.	.	NONE	NONE	. 10000
32	X1C2C4C1	LL	.	.	NONE	NONE	. 10000
33	X1C2C5C1	LL	.	.	NONE	NONE	.
34	X1C2C3C1	BS	.	.	NONE	NONE	.
35	X1C2C4C1	BS	.	.	NONE	NONE	.
36	X1C1C2C2	LL	.	.	NONE	NONE	.
37	X1C1C2C2	LL	.	.	NONE	NONE	.
38	X1C1C2C2	LL	.	.	NONE	NONE	.
39	X1C1C2C2	LL	.	.	NONE	NONE	.
40	X1C1C2C2	LL	.	.	NONE	NONE	. 10000
41	X1C1C2C2	LL	.	.	NONE	NONE	.
42	X1C1C2C2	LL	.	.	NONE	NONE	. 10000
43	X1C1C2C2	LL	.	.	NONE	NONE	.
44	X1C1C2C2	BS	2.500000	.	NONE	NONE	.
45	X1C1C2C2	BS	2.500000	.	NONE	NONE	.
46	X1C1C2C2	BS	2.500000	.	NONE	NONE	.
47	X1C1C2C2	BS	2.500000	.	NONE	NONE	.
48	X1C1C2C2	BS	2.500000	.	NONE	NONE	.
49	X1C1C2C2	BS	2.500000	.	NONE	NONE	. 10000
50	X1C1C2C2	LL	.	.	NONE	NONE	.
51	X1C1C2C2	LL	.	.	NONE	NONE	.
52	X1C1C2C2	LL	.	.	NONE	NONE	.
53	X1C1C2C2	BS	2.500000	.	NONE	NONE	.
54	X1C1C2C2	BS	2.500000	.	NONE	NONE	. 10000
55	X1C1C2C2	BS	2.500000	.	NONE	NONE	. 10000
56	X1C1C2C2	BS	2.500000	.	NONE	NONE	. 10000
57	X1C1C2C2	BS	2.500000	.	NONE	NONE	. 10000
58	X1C1C2C2	BS	2.500000	.	NONE	NONE	. 10000
59	X1C1C2C2	BS	2.500000	.	NONE	NONE	. 10000
60	X1C1C2C2	BS	2.500000	.	NONE	NONE	. 10000
61	X1C1C2C2	BS	2.500000	.	NONE	NONE	. 10000
62	X1C1C2C2	BS	2.500000	.	NONE	NONE	. 10000
63	X1C1C2C2	BS	2.500000	.	NONE	NONE	. 10000
64	X1C1C2C2	BS	2.500000	.	NONE	NONE	. 10000
65	X1C1C2C2	BS	2.500000	.	NONE	NONE	. 10000
66	X1C1C2C2	BS	2.500000	.	NONE	NONE	. 10000
67	X1C1C2C2	BS	2.500000	.	NONE	NONE	. 10000
68	X1C1C2C2	BS	2.500000	.	NONE	NONE	. 10000
69	X1C1C2C2	BS	2.500000	.	NONE	NONE	. 10000
70	X1C1C2C2	BS	2.500000	1.00000	NONE	NONE	.

A A A A

A A

A

A A

EXAMPLE PROBLEM, 3FD FUN FOR SUCCESSIVE SAT

SECTION 1 - FCWS

NUMBER	OR	...FCW...	AT	...ACTIVITY...	SLACK ACTIVITY	..LOWER LIMIT..	..UPPER LIMIT..	..DUAL ACTIVITY
A	1	01000000	BS	.22500	.22500-	ACNE	NONE	1.00000
A	2	01000000	BS
A	3	01000000	BS
A	4	01000000	BS
	5	01000000	BS
	6	01000000	BS
	7	01000000	BS
	8	01000000	BS
	9	01000000	BS
	10	01000000	BS
	11	01000000	BS
	12	01000000	BS
	13	01000000	BS
	14	01000000	BS
	15	01000000	BS
	16	01000000	BS
	17	01000000	BS
	18	01000000	BS
	19	01000000	BS
	20	01000000	BS
	21	01000000	BS
	22	01000000	BS
	23	01000000	BS
	24	01000000	BS
	25	01000000	BS
	26	01000000	BS
	27	01000000	BS
	28	01000000	BS
	29	01000000	BS

EXAMPLE PROBLEM, 2FC RUN FOR SUCCESSIVE SAT

SECTION 2 - COLUMNS

NUMBER	COLUMN	AT	...ACTIVITY...	..INPUT COST..	..LOWER LIMIT.	..UPPER LIMIT.	..REDUCED COST.
30	X1C1C1C1	BS	.	.	.	NONE	.
31	X1C1C2C1	BS	.	.	.	NONE	.
32	X1C1C3C1	LL	.	.	.	NONE	.C5000
33	X1C1C4C1	LL	.	.	.	NONE	.C5000
34	X1C1C5C1	BS	.	.	.	NONE	.
35	X1C1C6C1	BS	.	.	.	NONE	.
36	X1C1C7C2	LL	.	.	.	NONE	.
37	X1C1C8C2	LL	.	.	.	NONE	.
38	X1C1C9C2	LL	.	.	.	NONE	.C5000
39	X1C1C10C2	LL	.	.	.	NONE	.
40	X1C1C11C2	LL	.	.	.	NONE	.C5000
41	X1C1C12C3	BS	.	.	.	NONE	.
42	X1C1C13C3	BS	2.50000	.	.	NONE	.
43	X1C1C14C3	BS	2.50000	.	.	NONE	.
44	X1C1C15C3	BS	2.50000	.	.	NONE	.
45	X1C1C16C3	BS	2.50000	.	.	NONE	.
46	X1C1C17C3	BS	2.50000	.	.	NONE	.
47	X1C1C18C4	BS	2.50000	.	.	NONE	.
48	X1C1C19C4	LL	2.00000	.	.	NONE	.
49	X1C1C20C4	LL	.	.	.	NONE	.C5000
50	X1C1C21C4	BS	.	.	.	NONE	.
51	X1C1C22C4	BS	.	.	.	NONE	.
52	X1C1C23C4	BS	2.00000	.	.	NONE	.
53	X1C1C24C4	BS	2.00000	.	.	NONE	.
54	X1C1C25C4	BS	.	.	.	NONE	.
55	X1C1C26C5	BS	.	.	.	NONE	.
56	X1C1C27C5	BS	.	.	.	NONE	.C5000
57	X1C1C28C5	BS	.	.	.	NONE	.C5000
58	X1C1C29C5	BS	.	.	.	NONE	.
59	X1C1C30C5	BS	.	.	.	NONE	.
60	X1C1C31C5	BS	.	.	.	NONE	.
61	X1C1C32C5	BS	2.5000	.	.	NONE	.
62	X1C1C33C5	BS	.	.	.	NONE	.
63	X1C1C34C5	BS	.	.	.	NONE	.C5000
64	X1C1C35C5	BS	2.25000	.	.	NONE	.C5000
65	X1C1C36C5	BS	.	.	.	NONE	.
66	X1C1C37C5	BS	.	.	.	NONE	.
67	X1C1C38C5	BS	.	.	.	NONE	.
68	X1C1C39C5	BS	.	.	.	NONE	.C5000
69	X1C1C40C5	BS	.	.	.	NONE	.
70	ALPHA	BS	2.2500	1.00000	.	NONE	.

EXAMPLE PROBLEM, 41F FUN FOR SUCCESSIVE SAT

SECTION 1 - FLWS

NUMBER	OR	FLW	AT	ACTIVITY	SLACK ACTIVITY	LOWER LIMIT	UPPER LIMIT	EQUAL ACTIVITY
1	01	000000	US	.22500	.22500-	NONE	NONE	1.00000
2	01	000000	EE
3	01	000000	EE
4	01	000000	EE
5	01	000000	EE
6	01	000000	BS
7	01	000000	BS
8	01	000000	BS
9	01	000000	BS
10	01	000000	EE
11	01	000000	EE	5.00000	5.00000	5.00000	5.00000	1.00000-
12	01	000000	EE	1.00000-
13	01	000000	EE
14	01	000000	EE	2.00000	2.00000	2.00000	2.00000	1.00000-
15	01	000000	EE	1.00000-
16	01	000000	EE	1.00000-
17	01	000000	EE
18	01	000000	BS
19	01	000000	BS
20	01	000000	BS
21	01	000000	EE
22	01	000000	EE
23	01	000000	EE
24	01	000000	EE	2.50000	2.50000	2.50000	2.50000	1.00000
25	01	000000	EE	2.50000	2.50000	2.50000	2.50000	1.00000
26	01	000000	EE	2.25000	2.25000	2.25000	2.25000	1.00000
27	01	000000	EE	2.50000	2.50000	2.50000	2.50000	1.00000
28	01	000000	EE	2.50000	2.50000	2.50000	2.50000	1.00000
29	01	000000	EE	2.25000	2.25000	2.25000	2.25000	.

EXAMPLE PROBLEM, 4TH RUN FOR SUCCESSIVE SAT

SECTION 2 - COLUMNS

NUMBER	COLUMN	AT	...ACTIVITY...	..INPUT COST..	..LOWER LIMIT.	..UPPER LIMIT.	..REDUCED COST.
30	X1C1C1C1	BS	.	.	.	NONE	.
31	X1C2C2C1	BS	.	.	.	NONE	.10000
32	X1C3C4C1	LL	.	.	.	NONE	.10000
33	X1C2C5C1	LL	.	.	.	NONE	.
34	X1C2C3C1	BS	.	.	.	NONE	.
35	X1C1C4C1	BS	.	.	.	NONE	.
36	X1C1C2C2	LL	.	.	.	NONE	.
37	X1C1C3C2	LL	.	.	.	NONE	.10000
38	X1C1C3C2	LL	.	.	.	NONE	.
39	X1C1C4C2	LL	.	.	.	NONE	.
40	X1C1C4C2	LL	.	.	.	NONE	.
41	X1C1C2C3	LL	.	.	.	NONE	.
42	X1C1C1C3	BS	2.50000	.	.	NONE	.
43	X1C1C1C3	BS	2.50000	.	.	NONE	.
44	X1C1C4C3	BS	2.50000	.	.	NONE	.
45	X1C1C4C3	BS	2.50000	.	.	NONE	.
46	X1C1C4C3	BS	2.50000	.	.	NONE	.
47	X1C1C4C3	BS	2.50000	.	.	NONE	.
48	X1C1C4C4	LL	2.50000	.	.	NONE	.
49	X1C1C4C4	LL	2.50000	.	.	NONE	.10000
50	X1C1C2C4	LL	.	.	.	NONE	.
51	X1C1C2C4	BS	2.00000	.	.	NONE	.
52	X1C1C4C4	BS	.	.	.	NONE	.
53	X1C1C4C4	BS	.	.	.	NONE	.
54	X1C1C4C4	BS	.	.	.	NONE	.
55	X1C1C4C4	BS	.	.	.	NONE	.10000
56	X1C1C4C4	BS	.	.	.	NONE	.
57	X1C1C4C4	BS	.	.	.	NONE	.10000
58	X1C1C4C4	BS	.	.	.	NONE	.10000
59	X1C1C4C4	BS	.	.	.	NONE	.10000
60	X1C1C4C4	BS	.	.	.	NONE	.
61	X1C1C4C4	BS	2.50000	.	.	NONE	.
62	X1C1C4C4	BS	.	.	.	NONE	.
63	X1C1C4C4	BS	.	.	.	NONE	.10000
64	X1C1C4C4	BS	2.50000	.	.	NONE	.
65	X1C1C4C4	BS	.	.	.	NONE	.10000
66	X1C1C4C4	BS	.	.	.	NONE	.10000
67	X1C1C4C4	BS	.	.	.	NONE	.
68	X1C1C4C4	BS	.	.	.	NONE	.
69	X1C1C4C4	BS	2.25000	1.00000	.	NONE	.
70	ALPHA	BS	.	.	.	NONE	.

SECTION I - FCHS

NUMBER	...FCW...	AT	...ACTIVITY...	SLACK ACTIVITY	..LOWER LIMIT.	..UPPER LIMIT.	..DUAL ACTIVITY
1	01100201	BS	20000	20000-			1.00000
2	01100201	BS					
3	01100201	BS					
4	01100201	BS					
5	01100201	BS					
6	01100201	BS					
7	01100201	BS					
8	01100201	BS					
9	01100201	BS					
10	01100201	BS					
11	01100201	BS	5.00000		5.00000	5.00000	1.00000
12	01100201	BS					1.00000
13	01100201	BS					1.00000
14	01100201	BS	2.00000		2.00000	2.00000	1.00000
15	01100201	BS					1.00000
16	01100201	BS					
17	01100201	BS					
18	01100201	BS					
19	01100201	BS					
20	01100201	BS					
21	01100201	BS					
22	01100201	BS	2.50000		2.50000	2.50000	1.00000
23	01100201	BS	2.50000		2.50000	2.50000	1.00000
24	01100201	BS					
25	01100201	BS	2.50000		2.50000	2.50000	1.00000
26	01100201	BS	2.50000		2.50000	2.50000	1.00000
27	01100201	BS	2.50000		2.50000	2.50000	1.00000
28	01100201	BS	2.50000		2.50000	2.50000	1.00000
29	01100201	BS	2.50000		2.50000	2.50000	1.00000
30	01100201	BS	2.50000		2.50000	2.50000	1.00000

EXAMPLE PROBLEM, 5th RUN FOR SUCCESSIVE SAT

SECTION 2 - COLUMNS

NUMBER	COLUMN	AT	...ACTIVITY...	..INPUT CCST..	..LOWER LIMIT.	..UPPER LIMIT.	..REDUCED COST.
30	X1C1C1C1	BS	.	.	.	NONE	.
31	X1C2C2C1	BS	.	.	.	NONE	.10000
32	X1C3C2C1	LL	.	.	.	NONE	.10000
33	X1C4C2C1	LL	.	.	.	NONE	.
34	X1C1C3C1	BS	.	.	.	NONE	.
35	X1C2C3C1	BS	.	.	.	NONE	.10000
36	X1C3C3C1	LL	.	.	.	NONE	.10000
37	X1C4C3C1	LL	.	.	.	NONE	.10000
38	X1C1C1C2	LL	.	.	.	NONE	.
39	X1C2C1C2	LL	.	.	.	NONE	.10000
40	X1C3C1C2	LL	.	.	.	NONE	.10000
41	X1C4C1C2	LL	.	.	.	NONE	.10000
42	X1C1C1C3	BS	2.500000	.	.	NONE	.
43	X1C2C1C3	BS	2.500000	.	.	NONE	.
44	X1C3C1C3	BS	2.500000	.	.	NONE	.
45	X1C4C1C3	BS	2.500000	.	.	NONE	.
46	X1C1C2C3	BS	2.500000	.	.	NONE	.
47	X1C2C2C3	BS	2.500000	.	.	NONE	.
48	X1C3C2C3	BS	2.500000	.	.	NONE	.
49	X1C4C2C3	BS	2.500000	.	.	NONE	.10000
50	X1C1C3C4	LL	.	.	.	NONE	.
51	X1C2C3C4	LL	.	.	.	NONE	.
52	X1C3C3C4	LL	2.000000	.	.	NONE	.
53	X1C4C3C4	LL	2.000000	.	.	NONE	.
54	X1C1C1C4	BS	.	.	.	NONE	.10000
55	X1C2C1C4	BS	.	.	.	NONE	.10000
56	X1C3C1C4	BS	.	.	.	NONE	.10000
57	X1C4C1C4	BS	.	.	.	NONE	.10000
58	X1C1C2C4	LL	.	.	.	NONE	.
59	X1C2C2C4	LL	.	.	.	NONE	.
60	X1C3C2C4	LL	.	.	.	NONE	.10000
61	X1C4C2C4	LL	.	.	.	NONE	.10000
62	X1C1C3C5	BS	.	.	.	NONE	.
63	X1C2C3C5	BS	.	.	.	NONE	.10000
64	X1C3C3C5	LL	.	.	.	NONE	.10000
65	X1C4C3C5	LL	.	.	.	NONE	.
66	X1C1C1C6	LL	.	.	.	NONE	.
67	X1C2C1C6	LL	.	.	.	NONE	.
68	X1C3C1C6	LL	.	.	.	NONE	.
69	X1C4C1C6	LL	.	.	.	NONE	.10000
70	ALPHA	BS	2.000000	1.000000	.	NONE	.

EXAMPLE PROBLEM, 6TH RUN FOR SUCCESSIVE SAT

SECTION 1 - FCWS

NUMBER	...	FCW...	AT	...ACTIVITY...	SLACK ACTIVITY	...LOWER LIMIT...	...UPPER LIMIT...	...DUAL ACTIVITY
1	DE	02	BS	.	.	NONE	NONE	1.00000
2	CI	02	BS10000-
3	CI	03	BS10000-
4	CI	04	BS10000-
5	CI	05	BS
6	CI	06	BS
7	CI	07	BS
8	CI	08	BS
9	CI	09	BS
10	CI	10	BS
11	CI	11	BS	5.00000	5.00000	5.00000	5.00000	.
12	CI	12	BS
13	CI	13	BS	2.00000	2.00000	2.00000	2.00000	.
14	CI	14	BS
15	CI	15	BS10000-
16	CI	16	BS
17	CI	17	BS
18	CI	18	BS
19	CI	19	BS10000-
20	CI	20	BS10000-
21	CI	21	BS10000-
22	CI	22	BS	2.00000	2.00000	2.00000	2.00000	.
23	CI	23	BS	2.00000	2.00000	2.00000	2.00000	.
24	CI	24	BS	2.00000	2.00000	2.00000	2.00000	.
25	CI	25	BS	2.00000	2.00000	2.00000	2.00000	.
26	CI	26	BS	2.00000	2.00000	2.00000	2.00000	.
27	CI	27	BS	2.00000	2.00000	2.00000	2.00000	.
28	CI	28	BS	2.00000	2.00000	2.00000	2.00000	.
29	CI	29	BS	2.00000	2.00000	2.00000	2.00000	.
30	CI	30	BS	2.00000	2.00000	2.00000	2.00000	.

EXAMPLE PROBLEM, 61F FUN FOR SUCCESSIVE SAT

SECTION 2 - COLUMNS

NUMBER	COLUMN	AT	...ACTIVITY...	..INPUT COST..	..LOWER LIMIT.	..UPPER LIMIT.	..REDUCED COST.
30	X1C2C1C1	BS	.	.	.	NONE	.
31	X1C3C2C1	BS	.	.	.	NONE	.
32	X1C3C4C1	LL	.	.	.	NONE	10000
33	X1C2C3C1	LL	.	.	.	NONE	10000
34	X1C2C3C1	BS	.	.	.	NONE	.
35	X1C1C4C1	BS	.	.	.	NONE	.
36	X1C1C2C2	LL	.	.	.	NONE	.
37	X1C1C3C2	LL	.	.	.	NONE	.
38	X1C1C3C2	LL	.	.	.	NONE	.
39	X1C1C4C2	LL	.	.	.	NONE	10000
40	X1C1C4C2	LL	.	.	.	NONE	.
41	X1C1C4C3	BS	.	.	.	NONE	.
42	X1C1C4C3	BS	2.00000	.	.	NONE	.
43	X1C1C3C3	BS	2.00000	.	.	NONE	.
44	X1C1C4C3	BS	2.00000	.	.	NONE	.
45	X1C2C3C3	BS	2.00000	.	.	NONE	.
46	X1C2C3C3	BS	2.00000	.	.	NONE	.
47	X1C2C4C3	BS	2.00000	.	.	NONE	.
48	X1C1C4C4	BS	2.00000	.	.	NONE	.
49	X1C1C4C4	LL	.	.	.	NONE	10000
50	X1C1C3C4	LL	.	.	.	NONE	.
51	X1C1C3C4	LL	.	.	.	NONE	.
52	X1C1C4C4	BS	2.00000	.	.	NONE	.
53	X1C2C4C4	BS	2.00000	.	.	NONE	.
54	X1C2C4C4	BS	2.00000	.	.	NONE	.
55	X1C1C2C5	BS	.	.	.	NONE	10000
56	X1C1C2C5	LL	.	.	.	NONE	10000
57	X1C1C3C5	LL	.	.	.	NONE	.
58	X1C1C3C5	BS	.	.	.	NONE	.
59	X1C1C4C5	LL	.	.	.	NONE	.
60	X1C1C4C5	LL	.	.	.	NONE	.
61	X1C1C3C6	BS	.	.	.	NONE	10000
62	X1C1C3C6	LL	.	.	.	NONE	10000
63	X1C1C4C6	LL	.	.	.	NONE	.
64	X1C1C4C6	LL	.	.	.	NONE	.
65	X1C1C5C6	BS	.	.	.	NONE	10000
66	X1C1C5C6	LL	.	.	.	NONE	.
67	X1C1C5C6	LL	.	.	.	NONE	.
68	X1C1C5C6	BS	.	.	.	NONE	.
69	X1C1C5C6	BS	.	.	.	NONE	.
70	ALPHA	BS	.	1.00000	.	NONE	.

APPENDIX E

OPTIMIZATION PROGRAM, INPUT DATA MATRIX PICTURE AND OUTPUT FOR MAX-SLACK APPROACH

```
//CHANG JOB (1538,1808),'CHANG JOON WOONG',CLASS=A
/**
/**      MPSIII EXAMPLE PROBLEM
/**
//EXEC MSSMPS
//CPC.SYSIN DD *
PROGRAM ('ND')
INITIALZ
TITLE ('EXAMPLE PROBLEM, 2ND RUN FOR MAX-SLACK')
MOVE (XOBJ,'OBJ')
MOVE (XRHS,'INPUT')
MOVE (XDATA,'LINEQS')
MOVE (XPBNAME,'PROJECT')
CONVERT ('SUMMARY')
SETUP ('MAX')
BCDOUT
PICTURE
WHIZARD
PRIMAL
SOLUTION
EXIT
PEND

/*
//EXEC.SYSIN DD *
NAME LINEQS
ROWS
N OBJ
C1000201
C1000301
C1000401
C1000501
C1000102
C1000302
C1000402
C1000502
C1000103
C1000203
C1000403
C1000503
C1000104
C1000204
C1000304
C1000504
C1000105
C1000205
C1000305
C1000405
L1010200
L1020100
L1010300
L1030200
L1020400
L1020500
L1040300
L1050400
COLUMNS
X1020101 C1000201 1
X1020101 L1020100 1
X1030201 C1000201 -1
X1030201 C1000301 1
X1030201 L1030200 1
X1020401 C1000201 1
X1020401 C1000401 -1
```


X1020401	L1020400	1
X1020501	C1000201	1
X1020501	C1000501	-1
X1020501	L1020500	1
X1040301	C1000301	-1
X1040301	C1000401	1
X1040301	L1040300	1
X1050401	C1000401	-1
X1050401	C1000501	1
X1050401	L1050400	1
X1010202	C1000102	1
X1010202	L1010200	1
X1010302	C1000102	1
X1010302	C1000302	-1
X1010302	L1010300	1
X1030202	C1000302	1
X1030202	L1030200	1
X1040302	C1000302	-1
X1040302	C1000402	1
X1040302	L1040300	1
X1050402	C1000402	-1
X1050402	C1000502	1
X1050402	L1050400	1
X1010203	C1000103	1
X1010203	C1000203	-1
X1010203	L1010200	1
X1020103	C1000103	-1
X1020103	C1000203	1
X1020103	L1020100	1
X1010303	C1000103	1
X1010303	L1010300	1
X1020403	C1000203	1
X1020403	C1000403	-1
X1020403	L1020400	1
X1020503	C1000203	1
X1020503	C1000503	-1
X1020503	L1020500	1
X1040303	C1000403	1
X1040303	L1040300	1
X1050403	C1000403	-1
X1050403	C1000503	1
X1050403	L1050400	1
X1010204	C1000104	1
X1010204	C1000204	-1
X1010204	L1010200	1
X1020104	C1000104	-1
X1020104	C1000204	1
X1020104	L1020100	1
X1010304	C1000104	1
X1010304	C1000304	-1
X1010304	L1010300	1
X1030204	C1000204	-1
X1030204	C1000304	1
X1030204	L1030200	1
X1020404	C1000204	1
X1020404	L1020400	1
X1020504	C1000204	1
X1020504	C1000504	-1
X1020504	L1020500	1
X1050404	C1000504	1
X1050404	L1050400	1
X1010205	C1000105	1
X1010205	C1000205	-1
X1010205	L1010200	1
X1020105	C1000105	-1
X1020105	C1000205	1
X1020105	L1020100	1

X1010305	C1000105	1		
X1010305	C1000305	-1		
X1010305	L1010300	1		
X1030205	C1000205	-1		
X1030205	C1000305	1		
X1030205	L1030200	1		
X1020405	C1000205	1		
X1020405	C1000405	-1		
X1020405	L1020400	1		
X1020505	C1000205	1		
X1020505	L1020500	1		
X1040305	C1000305	-1		
X1040305	C1000405	1		
X1040305	L1040300	1		
S1000001	OBJ	1		L1010200 1
S1000002	OBJ	1		L1020100 1
S1000003	OBJ	1		L1010300 1
S1000004	OBJ	1		L1030200 1
S1000005	OBJ	1		L1020400 1
S1000006	OBJ	1		L1020500 1
S1000007	OBJ	1		L1040300 1
S1000008	OBJ	1		L1050400 1
ALPHA	OBJ	1		
ALPHA	L1010200	-0.100000E	02	
ALPHA	L1020100	-0.100000E	02	
ALPHA	L1010300	-0.100000E	02	
ALPHA	L1030200	-0.100000E	02	
ALPHA	L1020400	-0.100000E	02	
ALPHA	L1020500	-0.100000E	02	
ALPHA	L1040300	-0.100000E	02	
ALPHA	L1050400	-0.100000E	02	
RHS				
INPUT	C1000201	0		
INPUT	C1000301	0		
INPUT	C1000401	0		
INPUT	C1000501	0		
INPUT	C1000102	0		
INPUT	C1000302	0		
INPUT	C1000402	0		
INPUT	C1000502	0		
INPUT	C1000103	0		
INPUT	C1000203	5		
INPUT	C1000403	0		
INPUT	C1000503	0		
INPUT	C1000104	2		
INPUT	C1000204	0		
INPUT	C1000304	0		
INPUT	C1000504	0		
INPUT	C1000105	0		
INPUT	C1000205	0		
INPUT	C1000305	0		
INPUT	C1000405	0		
INPUT	L1010200	2.5		
INPUT	L1020100	2.5		
INPUT	L1010300	2.5		
INPUT	L1030200	2.5		
INPUT	L1020400	2.5		
INPUT	L1020500	2.5		
INPUT	L1040300	2.5		
INPUT	L1050400	2.5		
ENDATA				
/*				
//				

SECTION 1 - FCHS

69

EXAMPLE PROBLEM, 2ND RUN FOR SLACK-MAX

SECTION 2 - COLUMNS

NUMBER	COLUMN.	AT	...ACTIVITY...	..INPLT COST..	..LOWER LIMIT.	..UPPER LIMIT.	..REDUCED COST.
30	X1C3C1C1	BS	.	.	.	NONE	.
31	X1C3C2C1	BS	.	.	.	NONE	7.0000C-
32	X1C3C4C1	LL	.	.	.	NONE	4.0000C-
33	X1C3C5C1	LL	.	.	.	NONE	.
34	X1C3C5C1	BS	.	.	.	NONE	.
35	X1C3C4C1	BS	.	.	.	NONE	1.0000C-
36	X1C1C2C2	LL	.	.	.	NONE	2.0000C-
37	X1C1C2C2	LL	.	.	.	NONE	1.0000C-
38	X1C1C2C2	LL	.	.	.	NONE	1.0000C-
39	X1C1C2C2	LL	.	.	.	NONE	1.0000C-
40	X1C1C2C2	LL	.	.	.	NONE	2.0000C-
41	X1C1C2C3	LL	.	.	.	NONE	.
42	X1C1C1C3	BS	2.0000C	.	.	NONE	.
43	X1C1C1C3	BS	2.0000C	.	.	NONE	.
44	X1C1C4C3	BS	2.0000C	.	.	NONE	.
45	X1C1C5C3	BS	2.0000C	.	.	NONE	.
46	X1C1C5C3	BS	2.0000C	.	.	NONE	.
47	X1C1C2C4	BS	2.0000C	.	.	NONE	2.0000C-
48	X1C1C2C4	LL	.	.	.	NONE	2.0000C-
49	X1C1C2C4	LL	.	.	.	NONE	.
50	X1C1C2C4	BS	.	.	.	NONE	.
51	X1C1C2C4	BS	2.0000C	.	.	NONE	2.0000C-
52	X1C1C2C4	BS	2.0000C	.	.	NONE	.
53	X1C1C2C4	BS	2.0000C	.	.	NONE	2.0000C-
54	X1C1C2C4	BS	2.0000C	.	.	NONE	4.0000C-
55	X1C1C2C4	BS	2.0000C	.	.	NONE	4.0000C-
56	X1C1C2C4	BS	2.0000C	.	.	NONE	.
57	X1C1C2C4	BS	2.0000C	.	.	NONE	.
58	X1C1C2C4	BS	2.0000C	.	.	NONE	1.0000C-
59	X1C1C2C4	BS	2.0000C	.	.	NONE	1.0000C-
60	X1C1C2C4	BS	2.0000C	.	.	NONE	1.0000C-
61	X1C1C2C4	BS	2.0000C	.	.	NONE	1.0000C-
62	X1C1C2C4	BS	2.0000C	.	.	NONE	1.0000C-
63	X1C1C2C4	BS	2.0000C	.	.	NONE	1.0000C-
64	X1C1C2C4	BS	2.0000C	.	.	NONE	1.0000C-
65	X1C1C2C4	BS	2.0000C	.	.	NONE	1.0000C-
66	X1C1C2C4	BS	2.0000C	.	.	NONE	1.0000C-
67	X1C1C2C4	BS	2.0000C	.	.	NONE	1.0000C-
68	X1C1C2C4	BS	2.0000C	.	.	NONE	1.0000C-
69	X1C1C2C4	BS	2.0000C	.	.	NONE	1.0000C-
70	X1C1C2C4	BS	2.0000C	.	.	NONE	1.0000C-

APPENDIX F

LINK FLOW FOR 9/36 NETWORK

Table F.1. Link Flow for 9/36 Network (); MAX-SLACK

LINK (NODES)	DESTINATION NODE									TOTAL FLOW
	1	2	3	4	5	6	7	8	9	
1-2		0.5			(4) 3.5					(4) 4
2-1										
1-3					(4) 4					(4) 4
3-1										
1-8					(4) 3.75					(4) 3.75
8-1		0.5								0.5
1-9					(3) 3.75					(3) 3.75
9-1										
2-3					1.63					1.63
3-2		(1) 2.25								(1) 2.25
2-4					(4) 1.88					(4) 1.88
4-2		(4) 2.25								(4) 2.25
2-9										
9-2		(4) 4								(4) 4
3-4					1.88					1.88
4-3		1.75								1.75
3-5					(4) 3.75					(4) 3.75
5-3		(1) 0.5								(1) 0.5
4-5					(4) 3.75					(4) 3.75
5-4										
4-6										

LINK (NODES)	DESTINATION NODE									TOTAL FLOW
	1	2	3	4	5	6	7	8	9	
6-4		(4) 4								(4) 4
5-6										
6-5					(4) 4					(4) 4
5-7										
7-5		(1) 0.5			(3) 3.5					(4) 4
6-7										
7-6		(4) 4								(4) 4
6-8										
8-6					(4) 4					(4) 4
7-8		2.25								2.25
8-7					1.63					1.63
7-9		(4) 2.25								(4) 2.25
9-7					(3) 1.88					(3) 1.88
8-9		1.75								1.75
9-8					1.88					1.88
REQ-MAT					$R_1(5)=15, R_5(2)=9$					

APPENDIX G

LINK FLOW FOR 13/60 NETWORK

Table G.1. Link Flow for 13/60 Network (); MAX-SLACK

LINK (NODES)	DESTINATION NODE													TOTAL	
	1	2	3	4	5	6	7	8	9	10	11	12	13		FLOW
1-2															
2-1												(3.75) 3.75		(3.75) 3.75	
1-3						(0.25)								(0.25)	
3-1	(3.75) 3.33													(3.75) 3.33	
1-4												1.39		1.39	
4-1	(3.75) 3.33													(3.75) 3.33	
1-5												(3.75) 2.36		(3.75) 2.36	
5-1	(2.5) 3.33					(0.25)								(2.75) 3.33	
2-3												(3.75) 3.75		(3.75) 3.75	
3-2															
2-6												(3.75) 3.75		(3.75) 3.75	
6-2															
2-9												(3.75) 3.75		(3.75) 3.75	
9-2															

LINK (NODES)	DESTINATION NODE													TOTAL
	1	2	3	4	5	6	7	8	9	10	11	12	13	FLOW
3-4	0.73											1.88		2.61
4-3						1.56								1.56
3-6						(0.25) 1.97								(0.25) 1.97
6-3	(3.75) 2.92													(3.75) 2.92
3-7												(3.75) 1.88		(3.75) 1.88
7-3	1.15					0.41								1.56
4-5	0.97											1.39		2.36
5-4								2						2
4-7						0.64								0.64
7-4	2.6													2.6
4-8												1.89		1.89
8-4	0.97					0.2								1.17
5-8						(3.75) 2								(3.75) 2
8-5	(2.5) 2.36													(2.5) 2.36

L.LINK	DESTINATION NODE													TOTAL	
	(NODES)	1	2	3	4	5	6	7	8	9	10	11	12	13	FLOW
5-12													(3.75) 3.75		(3.75) 3.75
12-5															
6-7	0.95												(3.75) 1.97		(3.75) 2.92
7-6							(3.75) 1.97								(3.75) 1.97
6-9													0.47		0.47
9-6	1.9						0.07								1.97
6-10													1.3		1.3
10-6	(3.75) 1.97														(3.75) 1.97
7-8	0.73												(3.75) 1.88		(3.75) 2.6
8-7							(3.75) 1.17								(3.75) 1.17
7-10															
10-7	(3.75) 2.92														(3.75) 2.92
7-11													(3.75) 1.97		(3.75) 1.97
11-7	0.61						0.56								1.17

LINK (NODES)	DESTINATION NODE													TOTAL
	1	2	3	4	5	6	7	8	9	10	11	12	13	FLOW
8-11						0.63								0.63
11-8	(2.5) 2.6													(2.5) 2.6
8-12												(3.75) 3.75		(3.75) 3.75
12-8														
9-10								1.3						1.3
10-9		1.9												1.9
9-13												(3.75) 2.92		(3.75) 2.92
13-9						0.07								0.07
10-11	(2.5) 2.91													2.91
11-10														
10-13		0.3										2.62		2.92
13-10														
11-12												(3.75) 3.75		(3.75) 3.75
12-11														

LINK (NODES)	DESTINATION NODE													TOTAL
	1	2	3	4	5	6	7	8	9	10	11	12	13	FLOW
11-13						0.07								0.07
13-11	0.3											1.78		2.08
12-13														
13-12												(3.75) 3.75		(3.75) 3.75
REQ-MAT														

$R_2(12)=15$, $R_5(6)=4$, $R_{10}(1)=10$

APPENDIX H

LINK FLOW FOR SYMMETRIC REQ-MAT OF 13/60 NETWORK

Table H.1. Link Flow for Symmetric Req-Mat of 13/60 Network () ; MAX-SLACK

LINK	DESTINATION NODE													TOTAL
(NODES)	1	2	3	4	5	6	7	8	9	10	11	12	13	FLOW
1-2	(1/4)	1/2						(1/2)				(1/2)	(1/4) 1/4	(3/4) 3/4
2-1	1/4				1/2									(3/4) 3/4
1-3													(3/4) 3/4	(3/4) 3/4
3-1	(3/4) 3/4													(3/4) 3/4
1-4													(3/4) 3/4	(3/4) 3/4
4-1	(3/4) 3/4													(3/4) 3/4
1-5					1/2							(1/2)	(1/4) 1/4	(3/4) 3/4
5-1	(1/4) 1/4	1/2						(1/2)						(3/4) 3/4
2-3												(3/4) 3/4		(3/4) 3/4
3-2		(3/4) 3/4												(3/4) 3/4
2-6												(3/4) 3/4		(3/4) 3/4
6-2		(3/4) 3/4												(3/4) 3/4

LINK (NODES)	DESTINATION NODE												TOTAL FLOW
	1	2	3	4	5	6	7	8	9	10	11	12	13
2-9								(1/2)				(1/4) 1/2	(3/4) 3/4
9-2	(1/4) 1/4	(1/2)			1/2								(3/4) 3/4
3-4					(3/4) 3/4								(3/4) 3/4
4-3		(3/4) 3/4											(3/4) 3/4
3-6												(3/4) 3/4	(3/4) 3/4
6-3					(3/4) 3/4								(3/4) 3/4
3-7												(3/4) 3/4	(3/4) 3/4
7-3	(3/4) 3/4												(3/4) 3/4
4-5					(3/4) 3/4								(3/4) 3/4
5-4									(3/4) 3/4				(3/4) 3/4
4-7									(3/4) 3/4				(3/4) 3/4
7-4	(3/4) 3/4												(3/4) 3/4

LINK	DESTINATION NODE													TOTAL	
	(NODES)	1	2	3	4	5	6	7	8	9	10	11	12		13
4-8														(3/4) 3/4	(3/4) 3/4
8-4			(3/4) 3/4												(3/4) 3/4
5-8										(3/4) 3/4					(3/4) 3/4
8-5					(3/4) 3/4										(3/4) 3/4
5-12										1/2		(1/2)	(1/4) 1/4	(3/4) 3/4	
12-5		(1/4) 1/4	1/2		(1/2)									(3/4) 3/4	
6-7													(3/4) 3/4	(3/4) 3/4	
7-6										(3/4) 3/4				(3/4) 3/4	
6-9										(3/4) 3/4				(3/4) 3/4	
9-6					(3/4) 3/4									(3/4) 3/4	
6-10														(3/4) 3/4	
10-6			(3/4) 3/4										(3/4) 3/4	(3/4) 3/4	

LINK (NODES)	DESTINATION NODE												TOTAL FLOW
	1	2	3	4	5	6	7	8	9	10	11	12	13
7-8												(3/4) 3/4	(3/4) 3/4
8-7									(3/4) 3/4				(3/4) 3/4
7-10									(3/4) 3/4				(3/4) 3/4
10-7	(3/4) 3/4												(3/4) 3/4
7-11											(3/4) 3/4		(3/4) 3/4
11-7	(3/4) 3/4												(3/4) 3/4
8-11												(3/4) 3/4	(3/4) 3/4
11-8					(3/4) 3/4								(3/4) 3/4
8-12												(3/4) 3/4	(3/4) 3/4
12-8	(3/4) 3/4												(3/4) 3/4
9-10					(3/4) 3/4								(3/4) 3/4
10-9									(3/4) 3/4				(3/4) 3/4

LINK	DESTINATION NODE													TOTAL
(NODES)	1	2	3	4	5	6	7	8	9	10	11	12	13	FLOW
9-13					(1/2)							(1/4) 1/2	1/4	(3/4) 3/4
13-9	(1/4) 1/4	(1/2)							(1/2)					(3/4) 3/4
10-11					(3/4) 3/4									(3/4) 3/4
11-10		(3/4) 3/4												(3/4) 3/4
10-13												(3/4) 3/4		(3/4) 3/4
13-10	(3/4) 3/4													(3/4) 3/4
11-12												(3/4) 3/4		(3/4) 3/4
12-11	(3/4) 3/4													(3/4) 3/4
11-13													(3/4) 3/4	(3/4) 3/4
13-11	(3/4) 3/4													(3/4) 3/4
12-13		(1/2)											(1/4) 1/4	(3/4) 3/4
13-12	(1/4) 1/4				(1/2)				1/2					(3/4) 3/4
REQ-MAT													1/4	$R_1(13)=2, R_2(12)=2, R_5(9)=2, R_9(5)=2, R_{12}(2)=2, R_{13}(1)=2$

$$R_1(13)=2, R_2(12)=2, R_5(9)=2, R_9(5)=2, R_{12}(2)=2, R_{13}(1)=2$$

LIST OF REFERENCES

1. Arne, T., Computer Methods in Operations Research, p. 79, Academic Press, 1978.
2. Pooch, U. W., Green, W. H., and Moss, G. G., Telecommunications and Networking, Little, Brown Computer Systems Series, 1982.
3. Gass, S. I., Linear Programming Method and Application, 4th Ed., McGraw-Hill, 1975.
4. Kleinrock, L., Queueing Systems, V. 2, pp. 292-297, Wiley, 1976.
5. Davices, D. W., and Barber, D. A., Communication Networks for Computer, pp. 291-294, Wiley, 1973.
6. Carvis, H., Communications Network Analysis, pp. 103-104, Lexington, 1981.
7. Hiroshi, I. and Tadao, S., Theoretical Aspects in the Analysis and Synthesis of Packet Communication Networks, Proceedings of the IEEE, V. 66, No. 11, Nov 1978.
8. Kuo, F. F., Protocols and Techniques for Data Communications Networks, pp. 124-127, Prentice-Hall, 1981.
9. Ketron Inc., MPS III User Manual, 1980.
10. Phillips, D.T., and Alberto, G.D., Fundamentals of Network Analysis, Prentice Hall, 1981.

INITIAL DISTRIBUTION LIST

	No. Copies
1. Defense Technical Information Center Cameron Station Alexandria, VA 22314	2
2. Library, Code 0142 Naval Postgraduate School Monterey, CA 93943	2
3. Professor John W. Wozencraft, Code 62Wz Department of Electrical Engineering Naval Postgraduate School Monterey, CA 93943	3
4. Professor Paul H. Moose, Code 62Me Department of Electrical Engineering Naval Postgraduate School Monterey, CA 93943	2
5. Lieutenant Colonel Chang Joon Wong Korea MC Navy Headquarters (N-1) Seoul Korea	3

207817

Thesis

W86

Woong

c.1

An iteration algorithm for optimal network flows.

207817

Thesis

W86

Woong

c.1

An iteration algorithm for optimal network flows.

An iteration algorithm for optimal netwo



3 2768 000 98867 9

DUDLEY KNOX LIBRARY